BMS COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

A Technical Seminar Report based on review of Research Publication/Patent

SECURE HASH ALGORITHM

Submitted in partial fulfillment for the award of degree of

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:
YASHWANTH KIRAN S
1BM19CS187

Work carried out at

Internal Guide

Basavaraj Jakkali
Associate Professor
BMS College of Engineering

Department of Computer Science and Engineering
BMS College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019

2022-2023

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

I, YASHWANTH KIRAN S (1BM19CS187) student of 7thSemester, B.E, Department of Computer Science and Engineering, BMS College of Engineering, Bangalore, hereby declare that, this technical seminar entitled "SECURE HASH ALGORITHM" has been carried out under the guidance of **BASAVARAJ JAKKALI,** Associate Professor, Department of CSE, BMS College of Engineering, Bangalore during the academic semester October - February 2023. I also declare that to the best of my knowledge and belief, the technical seminar report is not from part of any other report by any other students.

Signature of the Candidate

YASHWANTH KIRAN S (1BM19CS187)

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Technical Seminar titled "**SECURE HASH ALGORITHM**" has been carried out by YASHWANTH KIRAN S (1BM19CS187) during the academic year 2022-2023.

Signature of the Guide

Signature of the Head of the Department

Signature of Examiners with date

1. Internal Examiner _____
2. External Examiner _____

Table of Contents

Abstract

The **Secure Hash Algorithms** are a family of cryptographic hash functions published by the National Institute of Standards and Technology (NIST) as a U.S. Federal Information Processing Standard (FIPS), including:

- **SHA-0**: A retronym applied to the original version of the 160-bit hash function published in 1993 under the name "SHA". It was withdrawn shortly after publication due to an undisclosed "significant flaw" and replaced by the slightly revised version SHA-1.
- **SHA-1**: A 160-bit hash function which resembles the earlier MD5 algorithm. This was designed by the National Security Agency (NSA) to be part of the Digital Signature Algorithm. Cryptographic weaknesses were discovered in SHA-1, and the standard was no longer approved for most cryptographic uses after 2010.
- **SHA-2**: A family of two similar hash functions, with different block sizes, known as *SHA-256* and *SHA-512*. They differ in the word size; SHA-256 uses 32-bit words where SHA-512 uses 64-bit words. There are also truncated versions of each standard, known as *SHA-224*, *SHA-384*, *SHA-512/224* and *SHA-512/256*. These were also designed by the NSA.
- **SHA-3**: A hash function formerly called *Keccak*, chosen in 2012 after a public competition among non-NSA designers. It supports the same hash lengths as SHA-2, and its internal structure differs significantly from the rest of the SHA family.

The corresponding standards are FIPS PUB 180 (original SHA), FIPS PUB 180-1 (SHA-1), FIPS PUB 180-2 (SHA-1, SHA-256, SHA-384, and SHA-512). NIST has updated Draft FIPS Publication 202, SHA-3 Standard separate from the Secure Hash Standard (SHS).

Chapter 1: Introduction

1.1 Overview

Secure Hash Algorithm (SHA) is a set of cryptographic hash functions designed by the National Security Agency (NSA) to protect against unauthorized access or tampering of data. The most commonly used versions of SHA are SHA-1, SHA-2, and SHA-3.

SHA functions take an input (often referred to as a "message") and produce a fixed-size output, known as a "hash" or "digest." The output is a unique representation of the input, and even small changes to the input will result in large changes to the output. This makes SHA functions useful for a variety of applications, including data integrity verification, digital signatures, and password hashing.

SHA-1 and SHA-2 are similar in structure and are both considered secure for most uses, although SHA-2 is considered more secure due to its larger output size. SHA-3, also known as Keccak, is the latest version of the SHA algorithm and is designed to provide additional security against potential weaknesses in SHA-1 and SHA-2.

SHA functions are widely used in various cryptographic protocols and standards, such as SSL/TLS, PGP, and SSH. They are also widely used in various industries, including finance, healthcare, and government.

In summary, Secure Hash Algorithm is a set of cryptographic hash functions that provide integrity and authenticity to the data, by generating a fixed size output called digest, that can be used to check any changes made to the original data.

1.2 Motivation

The motivation for this report on Secure Hash Algorithms (SHAs) is to provide an overview of the various algorithms in this category and their use in ensuring data integrity and security.

SHAs are widely used in a variety of applications such as digital signatures, file integrity verification, and password hashing.

This report aims to explain the technical details of these algorithms and their strengths and weaknesses, as well as provide examples of how they are used in practice.

Additionally, the report aims to provide an analysis of the current state of SHAs and their potential future developments.

Overall, the goal of this report is to provide a comprehensive understanding of SHAs and their role in securing digital information.

1.3 Objective

The main objective of Secure Hash Algorithms (SHAs) is to ensure data integrity and security. These algorithms are used to generate a fixed-size output, known as a hash or message digest, from an input of any size. The output, or hash, is a unique representation of the input data, and any change in the input data will result in a different hash value.

One of the primary uses of SHAs is in digital signatures, where the hash of a message is signed with a private key to provide authentication and non-repudiation. The recipient can then use the sender's public key to verify the signature and confirm that the message has not been tampered with.

Another common use of SHAs is in file integrity verification, where the hash of a file is calculated and stored separately from the file. Later, the hash can be recalculated and compared to the stored value to confirm that the file has not been modified.

SHAs are also commonly used in password hashing, where the hash of a password is stored instead of the password itself. This ensures that even if an attacker gains access to the stored hashes, they will not be able to determine the original passwords.

The main objectives of Secure Hash Algorithm (SHA) include:

1. **Data Integrity:** The primary objective of SHA is to ensure the integrity of data by generating a unique fixed-size output, known as a hash, for a given input. This allows the recipient of the data to verify that the data has not been tampered with or altered in transit.
2. **Authentication:** SHA can also be used for authentication purposes by generating a hash for a given input and then comparing it to a known hash value. This allows for the verification of the identity of the sender or the authenticity of the data.
3. **Data Security:** SHA provides a level of security by making it difficult for an attacker to tamper with the data or create a fake version of it. The output, or hash, is a fixed-length string of characters that is unique to the input, making it difficult to predict the output for a given input.
4. **Collision Resistance:** SHA is designed to be collision-resistant, meaning it is difficult to find two different inputs that produce the same output hash.
5. **Efficiency:** SHA is designed to be efficient in terms of both computation time and memory usage. It is designed to be fast and efficient for both software and hardware implementations.
6. **Versatility:** There are several versions of SHA, including SHA-1, SHA-2, and SHA-3, each with different strengths and weaknesses. This allows for the appropriate version of the algorithm to be used depending on the specific use case or security requirements.

Chapter 2: Literature Survey

[1] S. -H. Lee and K. -W. Shin, "An efficient implementation of SHA processor including three hash algorithms (SHA-512, SHA-512/224, SHA-512/256)," 2018 International Conference on Electronics, Information, and Communication (ICEIC), Honolulu, HI, USA, 2018, pp. 1-4, doi: 10.23919/ELINFOCOM.2018.8330578.

This paper describes a design of SHA processor implementing three hash algorithms of SHA-512, SHA-512/224 and SHA-512/256. The SHA processor generates digests of three different lengths with 512, 224, and 256 bits according to hash algorithms. It was designed that the initial hash values of SHA512/224 and SHA-512/256 were generated using SHA-512 and it was based on 32-bit datapath, resulting in an area-efficient implementation. The SHA processor designed with HDL was verified by FPGA implementation. The SHA processor synthesized with 0.18µm CMOS cell library occupies 27,368 gate equivalents (GEs) and can operate up to 185 MHz clock frequency. The SHA processor can be used for IoT security applications.

[2] F. E. De Guzman, B. D. Gerardo and R. P. Medina, "Enhanced Secure Hash Algorithm-512 based on Quadratic Function," 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM), Baguio City, Philippines, 2018, pp. 1-6, doi: 10.1109/HNICEM.2018.8666419.

This paper attempts to introduce the enhanced SHA-1 algorithm which features a simple quadratic function that will control the selection of primitive function and constant used per round of SHA-1. The message digest for this enhancement is designed for 512 hashed value that will answer the possible occurrence of hash collisions. Moreover, this features the architecture of 8 registers of A, B, C, D, E, F, G, and H which consists of 64 bits out of the total 512 bits. The testing of frequency for Q15 and Q0 will prove that the selection of primitive function and the constant used are not equally distributed. Implementation of extended bits for hash message will provide additional resources for dictionary attacks and the extension of its hash outputs will provide an extended time for providing a permutation of 512 hash bits.

[3] F. Kahri, H. Mestiri, B. Bouallegue and M. Machhout, "An efficient fault detection scheme for the secure hash algorithm SHA-512," 2017 International Conference on Green Energy Conversion Systems (GECS), Hammamet, Tunisia, 2017, pp. 1-5, doi: 10.1109/GECS.2017.8066141.

To protect the implementation of the standard Secure Hash Algorithm (SHA) against attacks. We have proposed a number of countermeasures. This paper present a proposed new fault detection scheme. It is based on the hybrid redundancy. The simulation results prove that the fault coverage achieves 99.999% for our scheme proposed. Also, our proposed detection scheme has been implemented on Xilinx Virtex-II Pro FPGA. It is fault coverage, area degradation, frequency, throughput and efficiency overhead have been compared and it is shown that the proposed scheme allows a trade-off between the security and the implementation cost of the SHA implementation.

[4] Revati Raman Dewanagn, Deepali Thombre and Kaushlendra Sharma in Implementation of Secure Hash Algorithm Using JAVA Programming[2] implemented SHA using Java Programming.Security is main issue when we talk about network , network based application and web based application or Web Services like SOAP(Simple object access protocol). Authentication of a user to learn his or her identity.In this paper the authors have implemented SHA (Secure Hash Algorithm), that is much capable to do this job for secure authentication of user.

[5] Adel, Radwa & Fouad, M. & Aboul-Dahab, Mohamed. (2013). DESIGN AND IMPLEMENTATION A NEW SECURITY HASH ALGORITHM BASED ON MD5 AND SHA-256. International Journal of Engineering Sciences & Emerging Technologies. 6. 29-36.

A cryptographic hash function has an important role in cryptography to achieve certain security goals such as authenticity, digital signatures, digital time stamping, and entity authentication. They are also strongly related to other important cryptographic tools such as block ciphers and pseudorandom functions .Due to the previous merits we present a proposal for a new secure hash algorithm based on the combination of some functions of SHA-256 (Secure Hash Algorithm 256) -with its message expansion modification- and MD5 (Message Digest 5) based on double-Davis-Mayer scheme to overcome the weakness existing in these functions. The proposal hash algorithm has been designed to satisfy the different levels of enhanced security and to resist the advanced hash attacks by increasing the complexity degree of the proposed hash algorithm. The security analysis of the proposed hash algorithm is compared to SHA256 and gives more security and highly acceptable results as shown in our security results and discussions

[6] Sahu, Aradhana & Ghosh, Samarendra. (2017). Review Paper on Secure Hash Algorithm With Its Variants. 10.13140/RG.2.2.13855.05289. The Cryptographic hash function is produce irreversible and unique hash value. It provides greater resistance against attack ,The variants of SHA algorithm are designed differently named are SHA-0, SHA-1, SHA-2, and SHA-3. This is a review paper which includes the comparisons between different secure hashing algorithms.

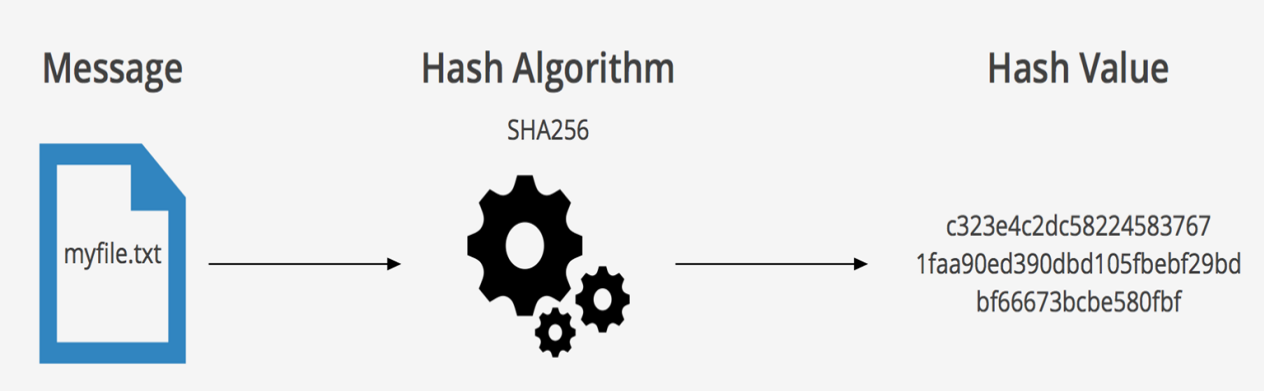
Chapter 3: Methodology/Techniques or Algorithm Used

Secure Hash Algorithm (SHA) works by taking an input (or "message") and processing it through a series of mathematical operations to produce a fixed-size output, known as a "hash" or "digest."

The specific steps of the algorithm vary depending on the specific version of SHA, but generally, the process can be broken down into the following steps:

1. The input message is padded to a specific length, so that its size is a multiple of a certain block size.
2. The padded message is then divided into fixed-size blocks, and processed through a series of logical and mathematical operations, such as bitwise operations, logical functions and modular additions. These operations are designed to be complex and non-reversible, so that even small changes to the input result in a completely different output.
3. The output of each block is then combined with the input of the next block, and the process is repeated for all blocks of the message.
4. Finally, the output of the last block is truncated to produce a fixed-size output (hash) that represents the input message.

- SHA-256 specifically takes a message of less than 2^64 bits in length and produces a 256-bit (32-byte) output.
- The output generated by the algorithm is designed to be unique to the input, meaning that even a small change to the input will result in a completely different output. This makes it useful for detecting tampering or modifications to data.



Chapter 4: Tools Used

- Visual Studio Code (VSCode) is a free and open-source code editor developed by Microsoft. It is available for Windows, Linux, and macOS.

  VSCode is a lightweight code editor that is designed to be easy to use, yet powerful and extensible.

- Cryptography module in python : Python provides a built-in module called "hashlib" that includes a variety of cryptographic hash functions, including SHA-256. It can be used to easily hash data in python.

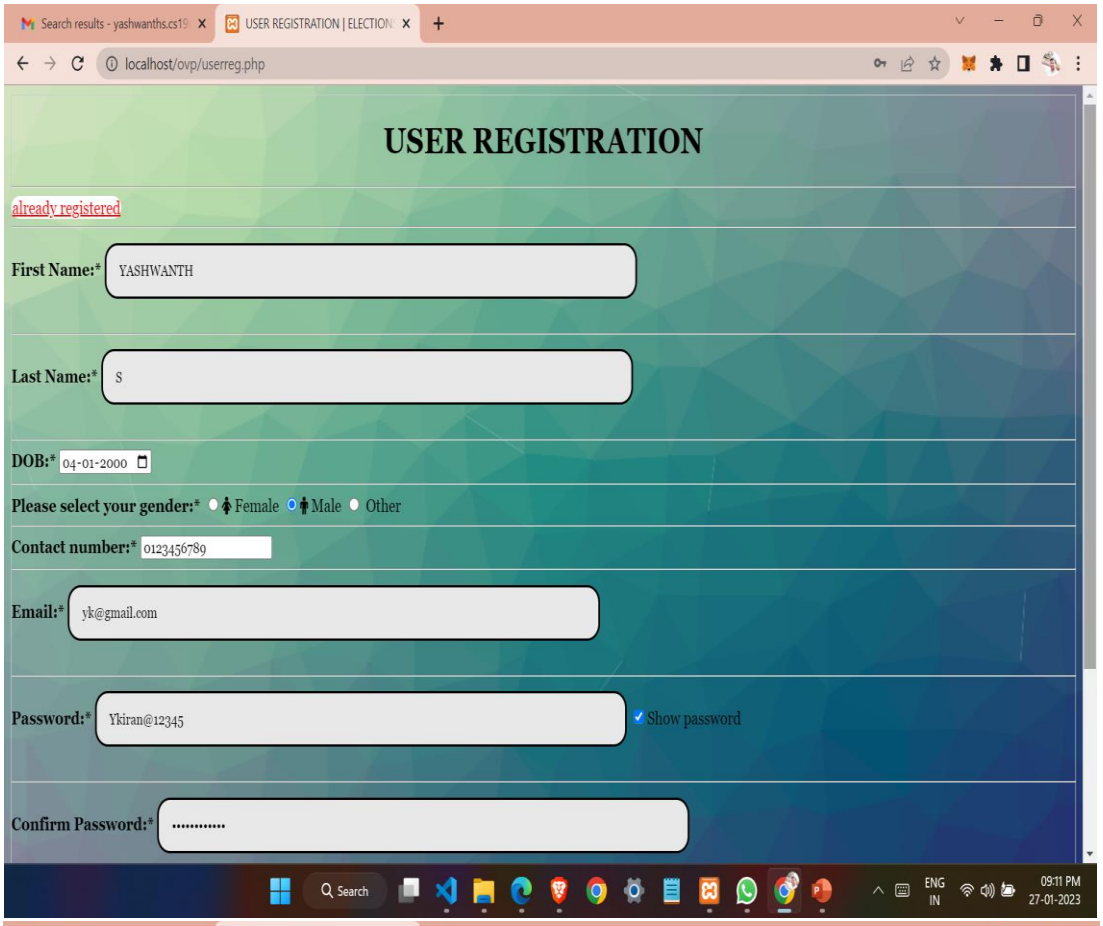
- XAMPP server. is a free and open-source cross-platform web server package that includes the Apache HTTP server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages.

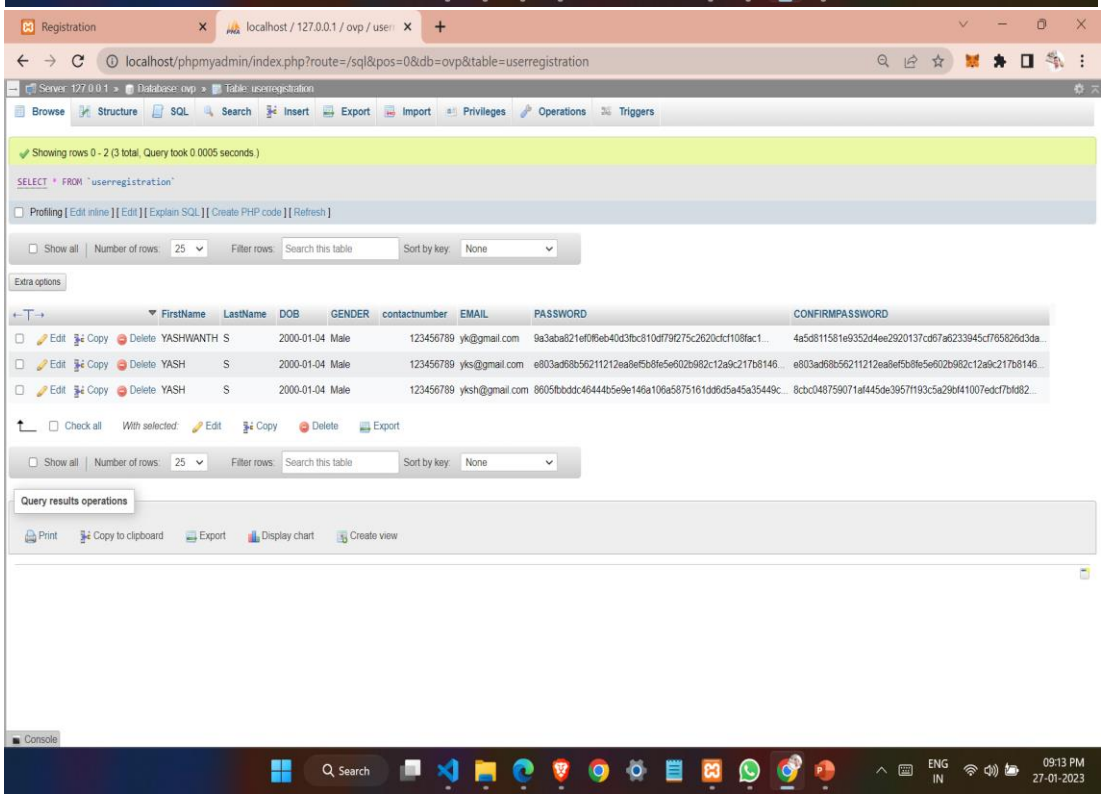
  It is commonly used for web development and testing on local computers. XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P).

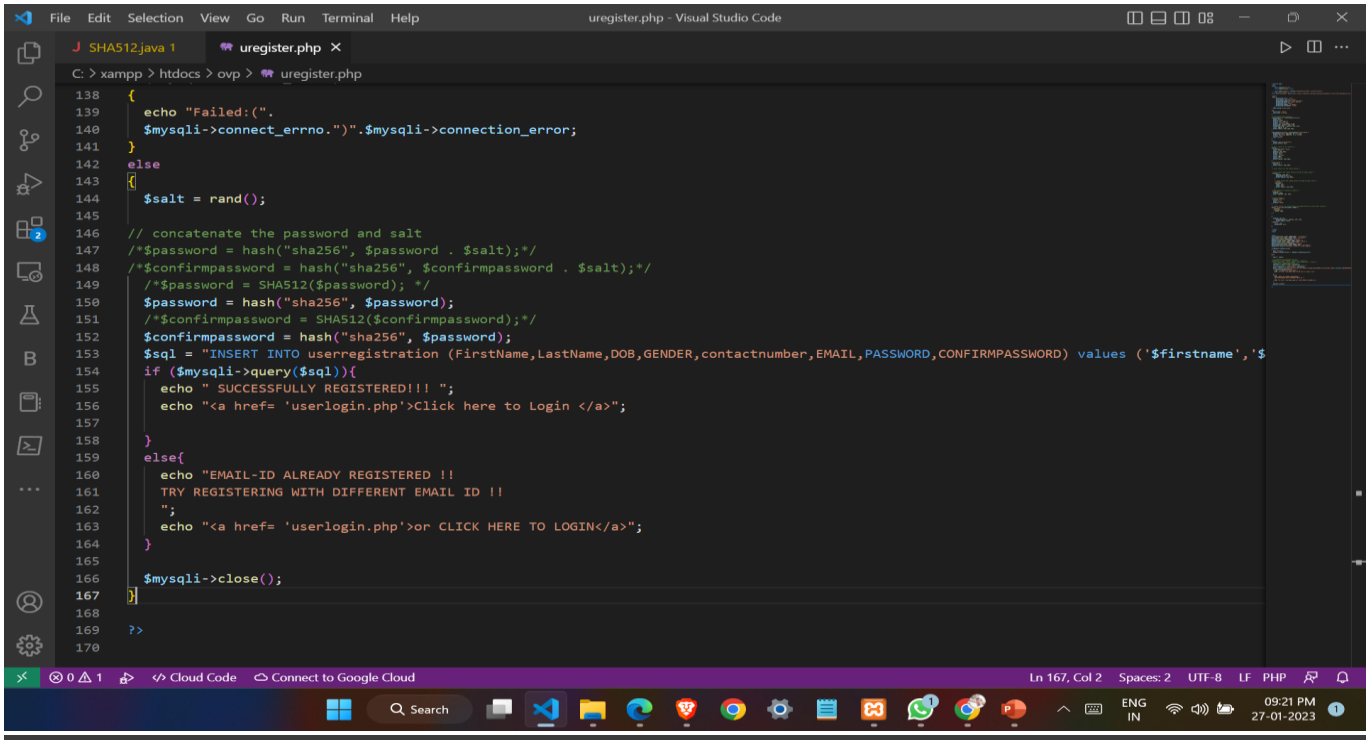
- Cryptool2 is a software tool for cryptography and cryptanalysis. It is a free and open-source program that allows users to encrypt and decrypt text, perform various cryptographic operations, and analyze and break ciphers.

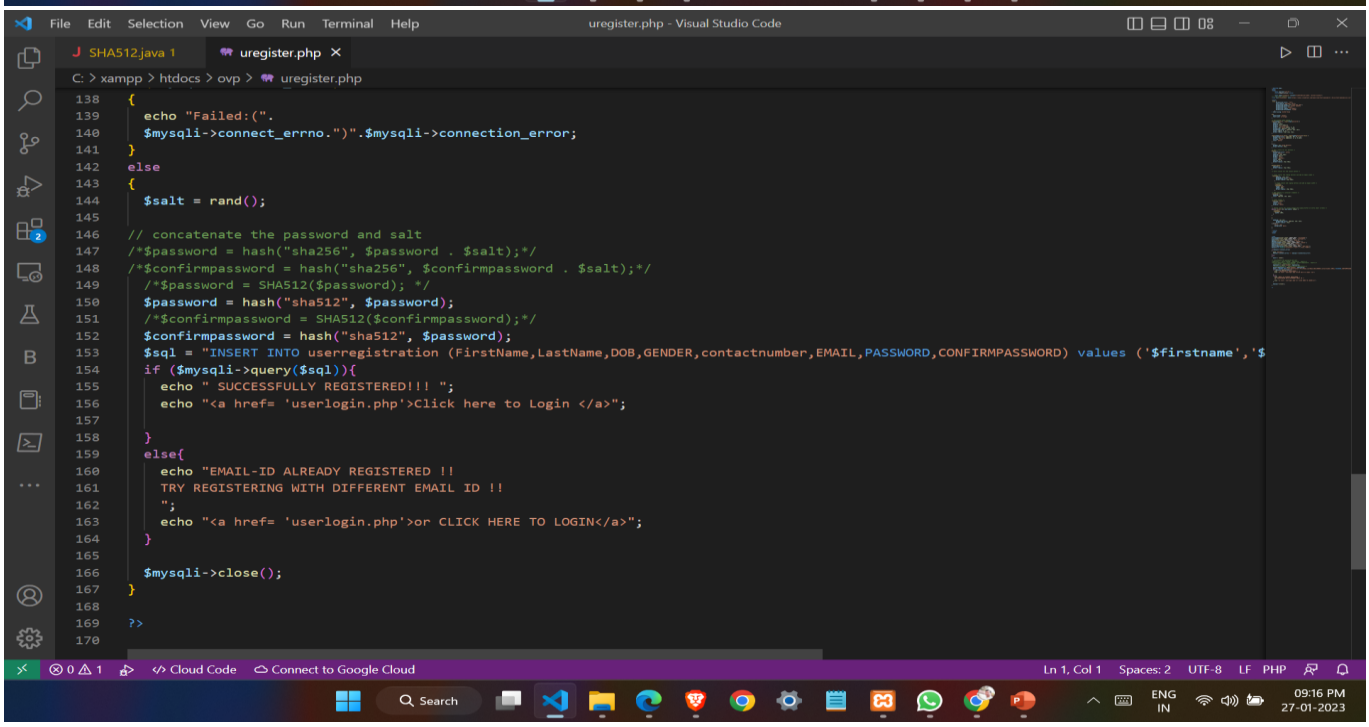
  Cryptool2 is designed for educational purposes and is commonly used in teaching and learning about cryptography. It is available for Windows and Linux operating systems. Cryptool 2 is a follow up tool to the original Cryptool, which was first release in 2001.

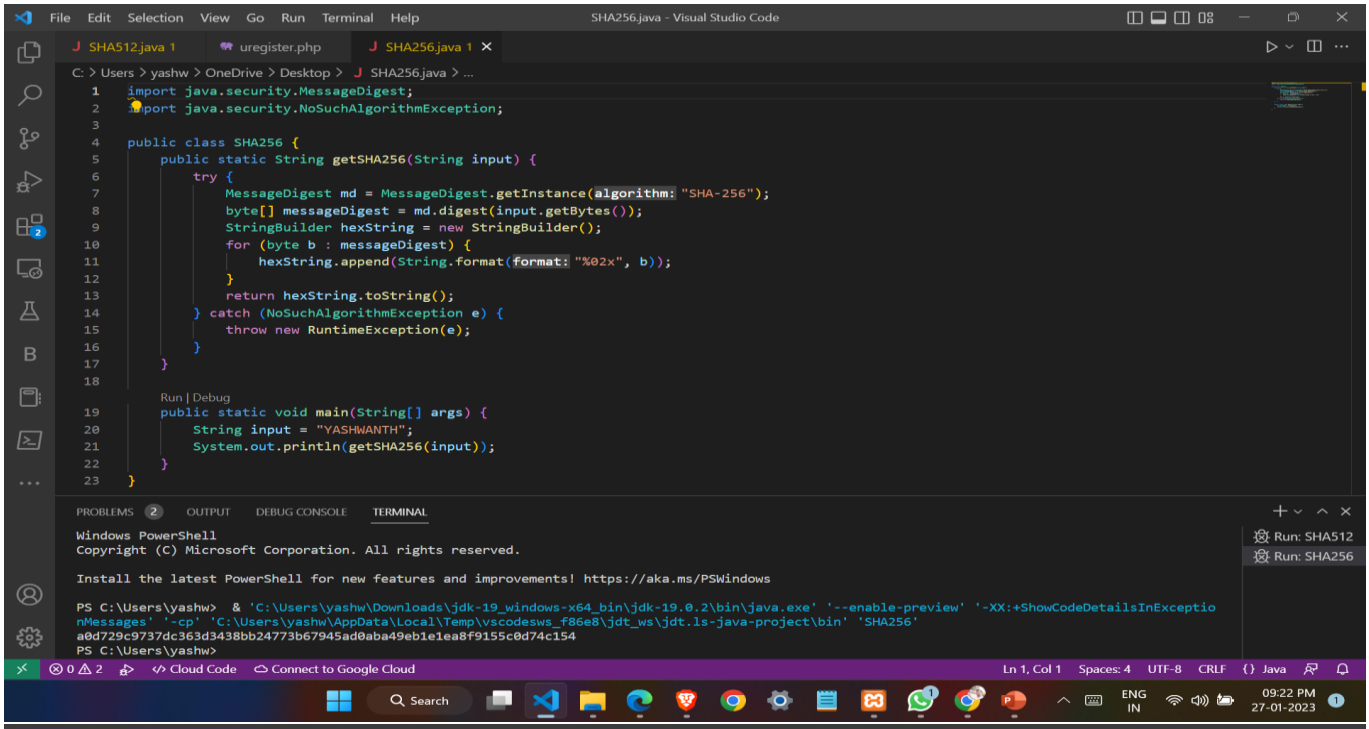
Chapter 5: Modules Implemented and Output

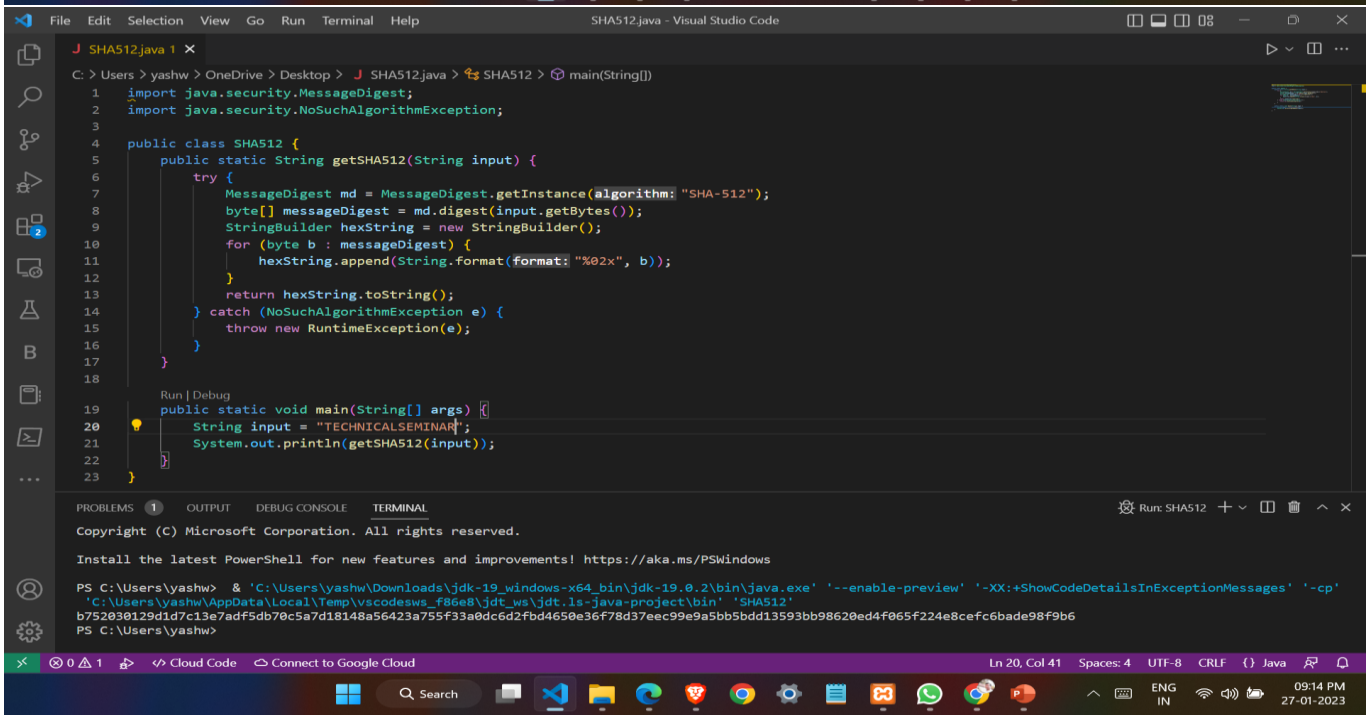





File Edit Selection View Go Run Terminal Help
uregister.php - Visual Studio Code
SHA512.java 1
uregister.php
C: > xampp > htdocs > ovp > uregister.php
{
echo "Failed:(".
$mysqli->connect_errno.")".$mysqli->connection_error;
}
else
{
$salt = rand();
// concatenate the password and salt
/*$password = hash("sha256", $password . $salt);*/
/*$confirmpassword = hash("sha256", $confirmpassword . $salt);*/
/*$password = SHA512($password); */
$password = hash("sha256", $password);
/*$confirmpassword = SHA512($confirmpassword);*/
$confirmpassword = hash("sha256", $password);
$sql = "INSERT INTO userregistration (FirstName,LastName,DOB,GENDER,contactnumber,EMAIL,PASSWORD,CONFIRMPASSWORD) values ('$firstname','$
if ($mysqli->query($sql)){
echo " SUCCESSFULLY REGISTERED!!! ";
echo "Click here to Login ";
}
else{
echo "EMAIL-ID ALREADY REGISTERED !!
TRY REGISTERING WITH DIFFERENT EMAIL ID !!
";
echo "or CLICK HERE TO LOGIN";
}
$mysqli->close();
}
?>
Cloud Code
Connect to Google Cloud
Ln 167, Col 2
Spaces: 2
UTF-8
LF
PHP
Search
ENG
IN
09:21 PM
27-01-2023


File Edit Selection View Go Run Terminal Help
uregister.php - Visual Studio Code
SHA512.java 1
uregister.php
C: > xampp > htdocs > ovp > uregister.php
{
echo "Failed:(".
$mysqli->connect_errno.")".$mysqli->connection_error;
}
else
{
$salt = rand();
// concatenate the password and salt
/*$password = hash("sha256", $password . $salt);*/
/*$confirmpassword = hash("sha256", $confirmpassword . $salt);*/
/*$password = SHA512($password); */
$password = hash("sha512", $password);
/*$confirmpassword = SHA512($confirmpassword);*/
$confirmpassword = hash("sha512", $password);
$sql = "INSERT INTO userregistration (FirstName,LastName,DOB,GENDER,contactnumber,EMAIL,PASSWORD,CONFIRMPASSWORD) values ('$firstname','$
if ($mysqli->query($sql)){
echo " SUCCESSFULLY REGISTERED!!! ";
echo "Click here to Login ";
}
else{
echo "EMAIL-ID ALREADY REGISTERED !!
TRY REGISTERING WITH DIFFERENT EMAIL ID !!
";
echo "or CLICK HERE TO LOGIN";
}
$mysqli->close();
}
?>
Cloud Code
Connect to Google Cloud
Ln 1, Col 1
Spaces: 2
UTF-8
LF
PHP
Search
ENG
IN
09:16 PM
27-01-2023


File Edit Selection View Go Run Terminal Help
SHA256.java - Visual Studio Code
SHA512.java 1
uregister.php
SHA256.java 1
C: > Users > yashw > OneDrive > Desktop > SHA256.java > ...
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class SHA256 {
public static String getSHA256(String input) {
try {
MessageDigest md = MessageDigest.getInstance(algorithm: "SHA-256");
byte[] messageDigest = md.digest(input.getBytes());
StringBuilder hexString = new StringBuilder();
for (byte b : messageDigest) {
hexString.append(String.format(format: "%02x", b));
}
return hexString.toString();
} catch (NoSuchAlgorithmException e) {
throw new RuntimeException(e);
}
}
Run | Debug
public static void main(String[] args) {
String input = "YASHWANTH";
System.out.println(getSHA256(input));
}
}
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\yashw> & 'C:\Users\yashw\Downloads\jdk-19_windows-x64_bin\jdk-19.0.2\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\yashw\AppData\Local\Temp\vscodesws_f86e8\jdt_ws\jdt.ls-java-project\bin' 'SHA256'
a0d729c9737dc363d3438bb24773b67945ad0aba49eb1e1ea8f9155c0d74c154
PS C:\Users\yashw>
Run: SHA512
Run: SHA256
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
Java
Search
ENG
IN
09:22 PM
27-01-2023


File Edit Selection View Go Run Terminal Help
SHA512.java - Visual Studio Code
SHA512.java 1
C: > Users > yashw > OneDrive > Desktop > SHA512.java > SHA512 > main(String[])
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
public class SHA512 {
public static String getSHA512(String input) {
try {
MessageDigest md = MessageDigest.getInstance(algorithm: "SHA-512");
byte[] messageDigest = md.digest(input.getBytes());
StringBuilder hexString = new StringBuilder();
for (byte b : messageDigest) {
hexString.append(String.format(format: "%02x", b));
}
return hexString.toString();
} catch (NoSuchAlgorithmException e) {
throw new RuntimeException(e);
}
}
Run | Debug
public static void main(String[] args) {
String input = "TECHNICALSEMINAR";
System.out.println(getSHA512(input));
}
}
PROBLEMS 1
OUTPUT
DEBUG CONSOLE
TERMINAL
Run: SHA512
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\yashw> & 'C:\Users\yashw\Downloads\jdk-19_windows-x64_bin\jdk-19.0.2\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\yashw\AppData\Local\Temp\vscodesws_f86e8\jdt_ws\jdt.ls-java-project\bin' 'SHA512'
b752030129d1d7c13e7adf5db70c5a7d18148a56423a755f33a0dc6d2fbd4650e36f78d37eec99e9a5bb5bdd13593bb98620ed4f065f224e8cefc6bade98f9b6
PS C:\Users\yashw>
Ln 20, Col 41
Spaces: 4
UTF-8
CRLF
Java
Search
ENG
IN
09:14 PM
27-01-2023

Chapter 6: Learnings and Takeaways from the Study

The **Secure Hash Algorithm (SHA)** is a widely used cryptographic hash function that is used to generate a fixed-size output, or digest, from a variable-size input, such as a message or file. The output, or digest, is a unique representation of the input and can be used to verify the integrity of the input data.

There are several different versions of the SHA algorithm, each with a different output size and level of security.

SHA-1 is the original version of the algorithm and has a 160-bit output size. It is now considered insecure and should not be used for new applications.

SHA-2 is a family of algorithms that includes SHA-224, SHA-256, SHA-384, and SHA-512. These algorithms have larger output sizes and offer stronger security than SHA-1.

SHA-3 is the latest version of the algorithm and was released in 2015. It includes five different algorithms, Keccak, Keccak-224, Keccak-256, Keccak-384, and Keccak-512, with output sizes of 224, 256, 384, and 512 bits respectively.

When using the SHA algorithm, it is important to use the latest version available to ensure the highest level of security. It is also important to use a unique and unpredictable salt value to prevent precomputation attacks.

In summary, Secure Hash Algorithm (SHA) is a widely used cryptographic hash function that generates a fixed-size output (digest) from a variable-size input. It is available in different versions like SHA-1, SHA-2, and SHA-3. It is considered important to use the latest version of SHA to ensure the highest level of security.

REFERENCES AND ANNEXURES

[1] F. E. De Guzman, B. D. Gerardo and R. P. Medina, "Implementation of Enhanced Secure Hash Algorithm Towards a Secured Web Portal," 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), Singapore, 2019, pp. 189-192, doi: 10.1109/CCOMS.2019.8821763.
[2] Sahu, Aradhana & Ghosh, Samarendra. (2017). Review Paper on Secure Hash Algorithm With Its Variants. 10.13140/RG.2.2.13855.05289.
[3] Adel, Radwa & Fouad, M. & Aboul-Dahab, Mohamed. (2013). DESIGN AND IMPLEMENTATION A NEW SECURITY HASH ALGORITHM BASED ON MD5 AND SHA-256. International Journal of Engineering Sciences & Emerging Technologies. 6. 29-36.
[4] F. E. De Guzman, B. D. Gerardo and R. P. Medina, "Enhanced Secure Hash Algorithm-512 based on Quadratic Function," 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM), Baguio City, Philippines, 2018, pp. 1-6, doi: 10.1109/HNICEM.2018.8666419.
[5] S. -H. Lee and K. -W. Shin, "An efficient implementation of SHA processor including three hash algorithms (SHA-512, SHA-512/224, SHA-512/256)," 2018 International Conference on Electronics, Information, and Communication (ICEIC), Honolulu, HI, USA, 2018, pp. 1-4, doi: 10.23919/ELINFOCOM.2018.8330578.
[6] F. Kahri, H. Mestiri, B. Bouallegue and M. Machhout, "An efficient fault detection scheme for the secure hash algorithm SHA-512," 2017 International Conference on Green Energy Conversion Systems (GECS), Hammamet, Tunisia, 2017, pp. 1-5, doi: 10.1109/GECS.2017.8066141.

[7] M. Kammoun, M. Elleuchi, M. Abid and M. S. BenSaleh, "FPGA-based implementation of the SHA-256 hash algorithm," 2020 IEEE International Conference on Design & Test of Integrated Micro & Nano-Systems (DTS), Hammamet, Tunisia, 2020, pp. 1-6, doi: 10.1109/DTS48731.2020.9196134.
[8] J. Wang, G. Liu, Y. Chen and S. Wang, "Construction and Analysis of SHA-256 Compression Function Based on Chaos S-Box," in IEEE Access, vol. 9, pp. 61768-61777, 2021, doi: 10.1109/ACCESS.2021.3071501.
[9] C. Jeong and Y. Kim, "Implementation of efficient SHA-256 hash algorithm for secure vehicle communication using FPGA," 2014 International SoC Design Conference (ISOCC), Jeju, Korea (South), 2014, pp. 224-225, doi: 10.1109/ISOCC.2014.7087617.
[10] H. Seta, T. Wati and I. C. Kusuma, "Implement Time Based One Time Password and Secure Hash Algorithm 1 for Security of Website Login Authentication," 2019 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS), Jakarta, Indonesia, 2019, pp. 115-120, doi: 10.1109/ICIMCIS48181.2019.8985196.